Claims

1. An application control system for a distributed computer system, including:

at least one node, each node including node controller means for starting, stopping and detecting a failure of a process on the node;

a plurality of application controllers wherein:

each application controller includes control means for managing at least one application according to an execution model; and

a first application controller including management means for managing a different type of software from a second application controller; and

an execution controller, the execution controller including execution control means for maintaining status information of processes started by the node controller executing on the at least one node and maintaining status and availability information of the at least one node.

2. The system of claim 1, including a plurality of nodes and wherein the execution controller means include means for maintaining status and availability information of the plurality of nodes.

3. The system of claim 1, including application controller means for initiating the creation of a container process.

4. An application control system for a distributed computer system, including:

at least one node, each node including a node controller configured to start, stop and detect a failure of a process on the node;

a plurality of application controllers wherein:

each application controller is configured to manage at least one application according to an execution model; and

a first application controller configured to manage applications according to an execution model that is different from the execution model of the applications managed by a second application controller; and

an execution controller, the execution controller configured to:

maintain status information of processes started by the node controller executing on the at least one node; and

maintain status and availability information of the at least one node.

5. The system of claim 4, including a plurality of nodes and wherein the execution controller maintains status and availability information of the plurality of nodes.

6. The system of claim 5, wherein a first application controller is on a first node and a second application controller is on a second node.

7. The system of claim 5, wherein the execution controller is on a node separate from any application controller.

8. The system of claim 5, wherein the execution controller is replicated on a plurality of nodes.

9. The system of claim 5, wherein at least one application controller is replicated on a plurality of nodes.

10. The system of claim 5, wherein at least one application controller includes logic configured to request the execution controller to start a process.

11. The system of claim 10, wherein the execution controller includes logic configured to request the node controller to start a process.

12. The system of claim 5, wherein the execution controller includes logic configured to request the node controller to start a process.

13. The system of claim 4, wherein an application controller includes logic configured to initiate the creation of a container process.

14. The system of claim 13, wherein the container process initiated by the application controller includes container controller logic configured to create at least one execution module.

15. The system of claim 14, wherein the logic to initiate the creation of a container process includes logic configured to send a message to the execution controller to cause the execution controller to inform the node controller to start the container process.

16. The system of claim 14, wherein the container process includes a first execution module from a first application and a second execution module from a second application.

17. The system of claim 14, wherein the container processes include a Java Virtual Machine and the execution units include Java classes.

18. The system of claim 14, wherein the container processes include an Enterprise JavaBeans Container class and the execution modules include Enterprise JavaBeans.

19. The system of claim 5, wherein the logic used by the execution controller to maintain status information of processes and maintain status and availability information of the nodes includes Enterprise JavaBeans.

20. The system of claim 5, wherein the at least one application controller includes Enterprise JavaBeans.

21. The system of claim 5, wherein the application controllers include distribution management logic to use distribution policy information as input and to output distribution information based on the distribution policy information.

22. The system of claim 21, wherein the application controller includes logic configured to invoke the distribution management logic upon at least one of a start of an application, a node failure, a process failure, a application failure, a node overload, and an addition of a node.

23. The system of claim 21, wherein the distribution information of the distribution management logic includes at least one of:

the number of container processes for an application,

the nodes on which each container process is located,

the number of execution modules to create;

the assignment of execution modules to container processes; and

the replication of execution modules.

24. The system of claim 23, wherein the application controllers include logic configured to retrieve distribution policy information for an application from an application definition.

25. The system of claim 23, wherein the application controllers include logic configured to receive distribution policy information for an application from a system administrator.

26. The system of claim 23, wherein the application controllers include logic configured to receive distribution policy information for an application from a configuration file.

27. The system of claim 21, wherein the distribution management logic is loaded from an application definition.

28. The system of claim 21, wherein the distribution management logic is loaded from a file separate from the application controller.

29. The application control system of claim 5, wherein a first application controller creates container processes only on a first node group.

30. The system of claim 21, wherein the execution controller includes logic configured to send a node failure notification message to each application controller with an application executing on a node upon a failure of the node.

31. The system of claim 21, wherein logic in a first node is configured to determine if the node including the execution module failed and upon such a determination the logic is configured to initiate an execution controller in the first node.

32. The system of claim 21, wherein the execution controller includes logic configured to send a process failure notification message to the application controller which initiated the creation of the failed process upon a failure of the process.

33. The system of claim 21, wherein the execution controller includes logic configured to send a process failure notification message to the application controller which initiated the creation of the failed process upon receipt of a process failure notification message from the node controller.

34. The system of claim 5, wherein at least one application controller manages a plurality of applications.

35. An application control system on a distributed computer system, including:

at least two nodes;

an application including at least one active execution module and at least one backup execution module;

at least two container processes each including at least one execution module; and

application controller logic wherein the application controller logic is configured to manage replication of an active execution module.

36. The system of claim 35, wherein the application controller logic includes distribution management logic configured to use distribution policy information as input and to output locations for at least one active and at least one backup execution module based on the distribution policy information.

37. The system of claim 36, wherein the application controller logic includes logic configured to invoke the distribution management logic upon at least one of a start of an application, a node failure, a process failure, an application failure, a node overload, and an addition of a node.

38. The system of claim 36, wherein the application controller logic includes a plurality of distribution management logics.

39. The system of claim 36, wherein the output of the distribution management logic includes at least one of:

the number of container processes for an application,
the nodes on which each container process is located,
the number of execution modules to create;
the assignment of execution modules to container processes; and
the replication of execution modules.

40. An application control system for a distributed computer system, including:

an application definition file including a definition of at least one execution module; and

an application controller module including distribution manager logic configured to distribute execution modules among a plurality of nodes based on node status information and replication constraint information.

41. An application control system for a distributed computer system, including:

an application definition file including a definition of at least one execution module; and

an application controller module including distribution manager logic configured to distribute container processes among a plurality of nodes and to distribute execution modules among the container processes based on at least two of node status information, container process status information and distribution policy information.

42. The system of claim 41, wherein the distribution policy information is retrieved from a policy file.

43. The system of claim 42, wherein the policy file is located within the application definition file.

44. The system of claim 41, wherein the distribution policy information is retrieved from a command line input.

45. The system of claim 41, wherein the application controller module retrieves the distribution manager logic from the application definition file.

46. The system of claim 41, wherein the application controller module retrieves the distribution manager logic from a file.

47. The system of claim 41, wherein the application controller module is located on a first node and a backup application controller is located on a second node.

48. The system of claim 41, further including a node controller module on a node configured to update the container process status information upon a failure of a container process on the node.

49. The system of claim 41, further including a node availability monitoring mechanism among at least two node controller modules configured to update the node status information upon a failure of a node in the distributed computer system.

50. The system of claim 41, wherein at least one container process includes execution modules from more than one application.

51. The system of claim 41, wherein at least one container process includes a Java Virtual Machine and the execution modules include at least one Java class.

52. The system of claim 41, wherein at least one container process includes an Enterprise JavaBean container class and at least one execution module includes Enterprise JavaBeans.

53. The system of claim 41, wherein the application controller module includes Enterprise JavaBeans.

54. The system of claim 41, wherein the node status information includes Enterprise JavaBeans.

55. The system of claim 41, wherein the application controller module includes logic configured to distribute at least one active execution module to a first node and one backup execution module to a second node, wherein the backup execution module is configured to be enabled upon the failure of the first execution module.

56. The system of claim 55, wherein the application controller module includes logic to enable the backup execution module.

57. The system of claim 41, wherein the application controller module further includes replication constraint information and the application controller module includes distribution manager logic configured to distribute execution modules among a plurality of nodes based on at least two of node status information, replication constraint information and distribution policy information.

58. The system of claim 41, wherein the execution modules include at least one server object and a client module including an object stub, wherein the object stub is configured to receive execution module location information from the application controller module.

59. The system of claim 58, wherein the client module includes a location cache module configured to store execution module location information, and wherein the client module may retrieve execution module location information from the cache.

60. The system of claim 59, wherein the application controller module is configured to update execution module location information in the cache of the client module.

61. The system of claim 60, wherein the application controller module keeps track of the client modules that have cached the location information.

62. The system of claim 58, wherein the client module includes logic configured to update execution module location information upon a failure of a node.

63. The system of claim 62, wherein the update of execution module location information is from the location of an active execution module to the location of a backup execution module.

64. The system of claim 62, wherein the application controller module keeps track of the client modules that have cached the location information.

65. The system of claim 62, wherein the update of execution module location information is performed upon the failure of an object stub attempting to access the active execution module.

66. The system of claim 62, wherein the update of execution module location information is performed upon the application controller module enabling a backup execution module corresponding to an active execution module whose location is included in the execution module location information of the client module.

67. The system of claim 41, wherein the application controller module includes partitioning information to partition server objects among execution modules.

68. The system of claim 67, wherein a client module includes a factory stub configured to initiate the creation of at least one server object in at least one execution module, the execution module to hold a server object is selected based on partitioning information received from the application controller module.

69. The system of claim 67, wherein a client module includes:

a partitioning information cache configured to store partition information received from the application controller module; and

a factory stub configured to initiate the creation of at least one server object in at least one execution module, the execution module to hold a server object is selected based on partitioning information contained in the partitioning information cache.

70. The system of claim 68, wherein the execution module selected to hold the server object is based on parameters sent from the client module to the factory stub and the partitioning information.

71. The system of claim 69, wherein the execution module selected to hold the server object is based on parameters sent from the client module to the factory stub and the partitioning information.

72. The system of claim 41, further including a client module including a factory stub configured to initiate creation of server objects, the factory stub further configured upon the receipt of an initiate creation of server object call to:

cause the application controller module to choose the execution module in which the server object will be created;

receive a response message from the application controller module after the execution controller module determines the location of the new server object; and

initiate the creation of the server object in the execution module specified by the response message.

73. The system of claim 72, wherein the application controller module creates a new execution module and includes the new execution module as the location for the server object in the response message sent to the factory stub.

74. A method of controlling execution of applications according to different models on a distributed computer system, including the steps of:

configuring a node controller module at least one node including a node to start, stop and detect a failure of a process on the node;

starting an execution controller module configured to maintain status information of processes started by the node controller module executing on the at least one node and maintain status and availability information of the at least one node; and

starting a plurality of application controllers module wherein:

each application controller module manages at least one application according to an execution model; and

a first application controller module manages applications according to an execution model that is different from the execution model of the applications managed by a second application controller module.

75. A method for starting an application on a distributed computer system, including the steps of:

indicating to an application controller module to start an application;

loading software including a distribution manager from a application definition file;

causing the distribution manager to select the at least one node on which a process providing the application will be located; and

indicating to a node controller module in the node selected by the distribution manager to start the process providing the application.

76. A method for starting an application control system for a distributed computer system, including the steps of:

starting a plurality of nodes;

starting a node controller module on each of the nodes;

sending heartbeat messages between the node controller modules;

electing a first node as a master node;

starting on the master node, an execution controller module configured to maintain node status information; and

starting a first and a second distinct application controller module configured to manage applications, upon a first and a second indication from the execution controller module.

77. A method for starting an application control system for a distributed computer system, including the steps of:

starting a plurality of nodes;

starting a node controller module on each of the nodes;

sending heartbeat messages between the node controller modules;

electing a first node as a master node;

starting on the master node, an execution controller module configured to maintain node status information;

electing at least a second node as master-backup nodes;

starting an execution controller module backup on each master-backup node; and

starting an application controller module, configured to manage applications, upon an indication from the execution controller module.

78. A method for executing applications on a distributed computer system, including the steps of:

indicating to an application controller module to begin an application by a system management tool sending an application start message to an application controller module;

creating a first container process on a first node and a second container process on a second node upon a message from the application controller module;

starting an active execution module including server objects in the first container process and a backup execution module including server objects in the second container process upon a message from the application controller module;

maintaining locations of the active execution module and the backup execution module in the application controller module;

monitoring the status of the active execution module and upon a failure of at least one active execution module reporting the failure to the application controller module;

promoting the backup execution module to active execution module status upon the application controller module receiving the message reporting the failure of the active execution module; and

updating the location of the active execution module and the backup execution module in the application controller module.

79. The method of claim 78, further including the step of notifying client modules of the change in location of the active execution module after the promoting step.

80. A method for starting an application including at least one execution module on a distributed computer system, including the steps of:

providing at least one distribution policy to an application controller module;

querying node status information to determine nodes available for executing the application;

creating container processes on at least one node based on the node status information and the distribution policy; and

creating execution modules in the container processes based on at least one of the distribution policy, the node status information and container process status information.

81. A method of distributing an application including a plurality of execution modules to a plurality of nodes, including the steps of:

associating a first execution module definition with a first name;

associating a second execution module definition second name;

associating a first group of nodes with the first name;

associating a second group of nodes with the second name;

creating execution modules from the first execution module definition in the first group of nodes; and

creating execution modules from the second execution module definition in the second group of nodes.

82. A method for recovering from a failure of a node in a distributed computer system, including the steps of:

detection of a failure of a first node by a second node;

updating node status information by the second node to indicate the failure of the first node;

indicating from the node status information to an application controller module the failure of the first node; and

invoking the distribution manager logic to recover from the failure of the first node.

83. The method of claim 82, wherein the detection is performed by the failure to receive a membership message by the second node.

84. The method of claim 82, wherein the invoking step includes creating a replacement execution module for an execution module previously located on the first node on a node different from the first node.

85. The method of claim 82, wherein the invoking step includes promoting a backup execution module, located on a node different from the first node, to an active execution module.

86. A method for recovery from a failure of a container process in a distributed computer system, including the steps of:

detection of a failure of a first container process by a node controller module;

indicating the failure of the first container process to an application controller module; and

invoking the distribution manager logic to recover from the failure of the first container process.

87. The method of claim 86, wherein the invoking step includes creating a replacement execution module for an execution module previously located in the first container process in a container process different from the first container process.

88. The method of claim 86, wherein the invoking step includes promoting a backup execution module, located in a container process different from the first container process, to an active execution module.

89. A method for recovery from a failure of an execution module in a distributed computer system, including the steps of:

detecting a failure of a first execution module by a container controller module;

indicating the failure of the first execution module by the container controller module to an application controller module; and

invoking the distribution manager logic to recover from the failure of the first execution module.

90. The method of claim 89, wherein the invoking step includes creating a replacement execution module for the first execution module.

91. The method of claim 89, wherein the invoking step includes promoting a backup execution module to an active execution module.

92. A method for adding a node to a distributed computer system, including the steps of:

starting a node controller module on an added node;

updating the node status information to indicate the addition of the added node to the distributed computer system;

indicating the addition of the added node to the application controller module from the node status information; and

invoking distribution management logic to distribute applications to the added node.

93. The method of claim 92, wherein the added node is added to at least one node group before invoking the distribution management logic.

94. A method for adding a node to a node group in a distributed computer system, including the steps of:

detecting if utilization of the nodes in a node group exceeds a predetermined threshold;

adding a first node to the node group upon detecting the exceeding of the predetermined threshold;

indicating the addition to the node group of the first node to the application controller module from the node group definition; and

invoking distribution management logic to distribute applications to the first node upon receipt of the indication that the first node was added to the node group.

95. The method of claim 94, wherein the adding of a first node to the node group is performed automatically by the application controller module.

96. The method of claim 94, wherein the adding of a first node to the node group is performed by a system administrator.

97. A method for removing applications from a node in a distributed computer system, including the steps of:

indicating to a node status information that a first node is blocked from the distributed computer system;

receiving an indication from the node status information that the first node is blocked by the application controller module, the application controller module including a distribution manager;

invoking the distribution manager upon receipt of the indication by the application controller module that the first node is blocked;

removing execution modules from the first node and creating replacement execution modules in nodes other than the first node by the distribution manager; and

indicating by the application controller module to the node status information that the removing of execution modules from the first node is complete.

98. A method for locating a server object in a distributed computer system, including the steps of:

invoking a programming language method associated with an operation on the server object by a client module on a stub,

obtaining by the stub the execution module in which the server object is located;

determining and obtaining by the stub whether container location information related to the execution module is located in the client module's location cache;

sending a request which invokes the operation on the server object from the stub to the execution module based on the obtained container location information;

if the container location information is determined to not be in the cache in the determining step then performing the following steps:

sending a request to the application controller module from the stub to determine the container in which the execution module is located;

receiving the container location of the execution module by the stub from the application controller module; and

storing the location of the container in which the execution module is located into the location cache.

99. The method of claim 98, wherein the server object is an Enterprise Java Bean object.

100. A method for creating a server object in a distributed computer system, including the steps of:

invoking a programming language method with at least zero parameters on a factory stub;

determining by the factory stub if a partition information related to the factory stub is located in a partition information cache on a client module of the factory stub and if the determination is that the partitioning information is not in the cache, retrieving the partitioning information from the application controller module and entering the partitioning information into the cache;

determining by the factory stub the execution module in which to create a server object by using at least one of the information in the partition information cache and the at least zero parameters;

passing a message from the factory stub to the execution module in which to create a server object to create the server object;

creating the server object in the execution module.

101. The method of claim 100, wherein the step of determining by the factory stub the execution module in which to create a server object includes using at least one of the information in the partition information cache, the at least zero parameters, and load balancing information.

102. The method of claim 100, wherein the server object is an Enterprise Java Bean object.

103. The method of claim 101, wherein the load balancing information is sent from the application controller module to the client module.

104. A method for creating a server object in a distributed computer system, including the steps of:

invoking a programming language method with at least zero parameters on a factory stub;

passing by the factory stub to the application controller module including server object type information and the at least zero parameters;

invoking the application controller module to determine the execution module in which to create the server object based on the server object type information and the at least zero parameters;

passing by the application controller module to the client module execution module information;

passing a message from the factory stub to the execution module in which to create a server object to create the server object; and

creating the server object in the execution module.

105. The method of claim 104, wherein the application controller module:

creates an execution module by passing a message to a container controller module; and

passes information related to the created execution module during the step of passing by the application controller module to the client module execution module information.

106. The method of claim 105, wherein the server object is an Enterprise Java Bean object.

107. A multiple-node distributed computer system, including:

a plurality of nodes connected over a communications network, the nodes including a node controller module configured to control a node on a distributed computer system;

at least one application controller module on at least one node; and

at least one execution controller on at least one node configured to maintain status information of processes started by the node controller executing on the nodes and maintain status availability information of at least one of the nodes.

108. The system of claim 107, wherein the execution controller is further configured to maintain status and availability of the nodes.

109. The system of claim 107, wherein the application controller is configured to initiate creation of container processes.

110. A computer system configured to communicate with a multiple-node system, including:

at least one application controller module configured to manage the execution, on the multiple-node system, of applications according to an execution model;

at least one node controller module configured to control the node; and

at least one execution controller configured to maintain status information of processes started by the node controller executing on the nodes and maintain status availability information of at least one of the nodes.

111. The system of claim 110, wherein the execution controller is further configured to maintain information relating to status and availability of the nodes in the multiple node system.

112. The system of claim 110, wherein the application controller is configured to initiate creation of container processes on the nodes of the multiple node system.

113. A computer-readable medium including instructions for performing a method when executed by a processor, for distributing an application including a plurality of execution modules to a plurality of nodes, the method including the steps of:

associating a first execution module definition with a first name;

associating a second execution module definition second name;

associating a first group of nodes with the first name;

associating a second group of nodes with the second name;

creating execution modules from the first execution module definition in the first group of nodes; and

creating execution modules from the second execution module definition in the second group of nodes.

114. A computer-readable medium including instructions for performing a method when executed by a processor, for controlling the execution of applications according to different models on a distributed computer system, the method including the steps of:

configuring a node controller module at least one node including a node to start, stop and detect a failure of a process on the node;

starting an execution controller module configured to maintain status information of processes started by the node controller module executing on the at least one node and maintain status and availability information of the at least one node; and

starting a plurality of application controllers module wherein each application controller module manages at least one application according to an execution model, and a first application controller module manages applications according to an execution model that is different from the execution model of the applications managed by a second application controller module.

115. An application control system for a distributed computer system, including:

a plurality of nodes, each node including a node controller configured to start, stop and detect a failure of a process on a node;

a plurality of application controllers wherein:

each application controller includes logic configured to manage at least one application according to an execution model, to initiate the creation of a container process and to use distribution policy information as input in order to generate distribution information output; and

a first application controller configured to manage applications according to an execution model that is different from the execution model of the applications managed by a second application controller, and wherein a first application controller is on a first node and a second application controller is on a second node; and at least one

application controller is replicated on a subset of the plurality of nodes; and

an execution controller, the execution controller configured to:

maintain status information of processes started by the node controller executing on at least one node;

request the node controller to start a process; and

maintain status and availability information of the plurality of nodes;

wherein the execution controller is on a node separate from any application controller and is replicated on a subset of the plurality of nodes.